CS 174a Template Instructions for Both Windows and Mac

XCode 6.1 OpenGL Project Setup:

1. Make a new **Command Line Tool** project, save it in the **same directory** that our template's folders reside in
2. Right-click main.cpp and **delete** it (move to trash)
3. Right-click on your project, select **"Add files"** to it, navigate to the **"my code"** folder and add **anim.cpp**
4. Do that again, this time highlighting and adding all the files from the **"CS174a Template"** folder
5. Click your project, click **Build Phases** tab, choose **Link Binary with Libraries**, hit the **+**, type **OpenGL** and add the framework that comes up, then do another and type **GLUT** and add that it too
6. Choose **"Edit Scheme"** (Command Key + Shift + Comma). In the **Run** tab, select **"Use custom working directory"** and type **$PROJECT_DIR/../exe** then **Compile** your project to make sure the whole thing works.

You're set up; now in **anim.cpp** you can insert your main code into **display()**, any helper functions above it, any other changes you decided for that file's other glut callbacks, and your extra shapes into **Shapes.h**.

Visual Studio 2013 OpenGL Project Setup:

1. Press **Ctrl+Shift+N** (new project). Select **empty project**. **Name it**.

2. Press **Ctrl+Alt+L**, then right-click the **Solution icon** and say "**Open Folder In File Explorer**" to navigate to the Solution folder that got created. Into there, **paste** the whole group of files we've provided.

3. Back in Visual Studio, right-click **Project icon** and click **Properties**. Fill in the following fields:

1. Select **"All Configurations"** at the top.

2. Configuration Properties > General > Output Directory:

..\Exe ($(Platform) $(Configuration))

Configuration Properties > General > Intermediate Directory:

Build ($(Platform) $(Configuration))

Build Events > Post-Build Event > Command Line:

xcopy "..\GL\$(Platform)*.dll" "$(OutDir)" /i /r /y

4. Press **Ctrl+Shift+A** (add existing item), navigate to the **"my code"** folder from the ones we pasted, and choose **anim.cpp**. Compile it to make sure the whole template works.

You're set up; now in **anim.cpp** you can insert your main code into **display()**, any helper functions above it, any other changes you decided for that file's other glut callbacks, and your extra shapes into **Shapes.h**.

Extra WebGL Instructions for Windows:

To convert your working C++ program into a .html:

1. Install Emscripten from the internet.

2. In the "my code" folder, click "emcmdprompt.bat".

3. To produce the WebGL page enter the following command, **replacing the texture image filenames with your own** if necessary:

emcc anim.cpp -o hello.html -std=c++11 --embed-file vshader.glsl --embed-file fshader.glsl --embed-file challenge.tga --embed-file earth.tga

Or to have it spend some extra time generating a smaller web page file that runs must faster, approaching C++ performance:

emcc anim.cpp -o hello.html -std=c++11 --embed-file vshader.glsl --embed-file fshader.glsl --embed-file challenge.tga --embed-file earth.tga -Oz -O3 --memory-init-file 0

Extra WebGL Instructions for Mac:

1. Download the portable Emscripten library (a folder).
2. In terminal, type `python2 --version`. If you get a "command not found", type the following:

```
cd /usr/bin
sudo ln python python2
sudo ln ../../System/Library/Frameworks/Python.framework/Versions/2.7/bin/python2.7 python22.7
```

Enter `python2 --version` again. It should now print Python 2.7.2

3. Use **cd** to navigate to your emsdk-portable folder and run this:

```
./emsdk update
./emsdk install latest
./emsdk activate latest
source ./emsdk_env.sh
```

4. Use **cd** to navigate to your **"my code"** folder. To produce the WebGL page enter the following command, **replacing the texture image filenames with your own** if necessary:

   ./../emsdk_portable/emscripten/1.29.0/emcc anim.cpp -o hello.html -std=c++11 --embed-file vshader.glsl --embed-file fshader.glsl --embed-file challenge.tga --embed-file earth.tga

   Or to have it spend some extra time generating a smaller web page file that runs must faster, approaching C++ performance:

   ./../emsdk_portable/emscripten/1.29.0/emcc anim.cpp -o hello.html -std=c++11 --embed-file vshader.glsl --embed-file fshader.glsl --embed-file challenge.tga --embed-file earth.tga -Oz -O3 --memory-init-file 0